Programación I - Grado de Ingeniería Informática y Doble Grado II-ADE - ETSIINF - UPM

Curso 2016/2017

Eval2 - SOLUCIÓN

10/01/2017

Apellidos:	Nombre (*rellenar con letra clara*):	
Grupo:	DNI:	Grado(II/ADE):

Ejercicio 1

Dado un array de naturales (int) con las horas que han trabajado los empleados de una empresa en un periodo de tiempo, define en Java la función *algunoPasa* especificada a continuación. **(2 puntos)**

```
/*
   FUNCIÓN algunoPasa (Natural[] horas, Natural n) --> Booleano
   POST: Determina si algún empleado ha trabajado más (>) de n horas.
   EJEMPLOS: (*)
     algunoPasa([], 8) = false
     algunoPasa([5], 35) = false
     algunoPasa([40, 35], 40) = false
     algunoPasa([5, 9, 41], 40) = true
*/
```

(*) Los arrays de los ejemplos se han denotado informalmente poniendo sus elementos entre corchetes.

```
static boolean algunoPasa (int[] horas, int n)
{
  boolean resultado = false;
  int i = 0;
  while (i < horas.length && !resultado)
  {
    if (horas[i] > n) {
      resultado = true;
    }
    else {
      i = i + 1;
    }
  }
  return resultado;
}
```

Define en Java las pruebas correspondientes a los ejemplos. **(0,5 puntos)**

```
static boolean prueba1algunoPasa = algunoPasa(new int[]{}, 8) == false;
static boolean prueba2algunoPasa = algunoPasa(new int[]{5}, 35) == false;
static boolean prueba3algunoPasa = algunoPasa(new int[]{40, 35}, 40) == false;
static boolean prueba4algunoPasa = algunoPasa(new int[]{5, 9, 41}, 40) == true;
```

Ejercicio 2

Dadas las siguientes funciones

```
static int igneo (int a, int b) {
  if (a >= b)
    return a;
  else
    return b;
}
static char ignoto (char[] algo)
{
  char resultado = algo[0];
  int i = 0;
  while (i <= algo.length-2)
  {
    resultado = (char)igneo(algo[i], algo[i+1]);
    i = i + 1;
  }
  return resultado;
}
```

Indica cuál es el valor devuelto por *ignoto* si recibe como argumento:

a) un array formado por la primera letra de tu nombre de pila.

b) un array formado por las dos primeras letras de tu nombre de pila

c) un array formado por las letras de tu nombre y tu primer apellido, sin espacios en blanco entre ellos y con todas las letras en minúsculas.

Utiliza la notación de corchetes para representar informalmente los arrays y ponlos en el espacio que se te deja para ello dentro de los paréntesis. Pon el resultado a la derecha de la flecha. **(0,5 puntos)**
Ejemplo:

> Para una persona llamada *Juan Li*, tendremos
> ignoto(['j', 'u', 'a', 'n','l', 'i']) –> ...

```
a) ignoto (['j']) --> 'j'
b) ignoto (['j', 'u']) --> 'u'
c) ignoto (['j', 'u', 'a', 'n','l', 'i']) --> 'l'
```

Completa la especificación de la función *ignoto*. **(0,5 puntos)**

```
/*
 * FUNCIÓN ignoto (Caracter[] algo)--> Caracter
 * PRE: longitud(algo) >= 1
 * POST: Si la longitud de algo es 1, devuelve el único elemento de algo.
 *       Si la longitud de algo es mayor o igual a 2,
 *       devuelve el mayor de entre el penúltimo y el ultimo de algo.
```

Ejercicio 3

Se quiere desarrollar la función *rodaja* especificada a continuación.

```
/*
  FUNCION rodaja (Entero[] col, Natural inicio, fin) --> Entero[]
  PRE: inicio, fin IN [0, longitud(col)-1]
  POST: resultado es la subcoleccion de numeros de "col" que esta
        entre las posiciones "inicio" y "fin", ambas incluídas.
 */
```

Completa los siguientes ejemplos (los arrays se han denotado informalmente poniendo sus elementos entre corchetes): **(1 punto)**

```
rodaja([5], 0, 0) = [5]
rodaja([7, 5], 0, 1) = [7, 5]
rodaja([6, 9, 0], 1, 2) = [9, 0]
rodaja([4, 8, 3, 6], 1, 2) = [8, 3]
rodaja([9, 0, 1, 4], 2, 1) = []
```

Define la función *rodaja* en Java (*). **(2 puntos)**
(*) No se puede utilizar las operaciones predefinidas en la clase *Arrays* de Java.

```
static int[] rodaja (int[] col, int inicio, int fin)
{
  int dimension = fin-inicio+1;
  if (dimension < 0) dimension = 0;
  int[] resultado = new int[dimension];
  for (int i = 0; i < dimension; i++)
  {
    resultado[i] = col[inicio+i];
  }
  return resultado;
}
```

Ejercicio 4

Se quiere desarrollar una aplicación para evaluar cuestionarios de tipo test, en los que cada respuesta viene dada como un carácter que representa la respuesta elegida ('a', 'b', 'c', ...).
Define en Java la función *puntosPregunta* especificada a continuación. **(1 punto)**

```
/* FUNCIÓN puntosPregunta (Caracter ru, rc) --> Entero
   POST: Dada una respuesta del usuario "ru" y una
         clave de respuesta correcta "rc", asigna 2 puntos
         si "rc" y "ru" coinciden y 0 puntos en caso contrario.
   EJEMPLOS:
     puntos('a', 'b') = 0
     puntos('a', 'a') = 2
     puntos('a', 'c') = 0
*/
```

```
static int puntosPregunta (char ru, char rc)
{
  if (ru == rc)
    return 2;
  else
    return 0;
}
```

Define en Java la función *puntosCuestionario* especificada a continuación. **(2,5 puntos)**

```
/* FUNCIÓN puntosCuestionario (Caracter[] respuestas,
                               Caracter[] claves) --> Entero
   PRE: longitud(respuestas) = longitud(claves)
   POST: Calcula el total de puntos alcanzado en un cuestionario.
         "respuestas" y "claves" contienen respectivamente
         las respuestas que ha dado el usuario y las respuestas
         correctas.
         Los puntos de cada pregunta vienen dados por la función
         "puntosPregunta".
   EJEMPLOS:
     puntosCuestionario(['a', 'b'], ['c', 'a']) = 0
     puntosCuestionario(['a', 'a', 'a'], ['a', 'b', 'a']) = 4
     puntosCuestionario(['a', 'b', 'b', 'c'], ['a', 'a', 'a', 'b']) = 2
*/
```

```
static int puntosCuestionario (char[] respuestas, char[] claves)
{
  int puntos = 0;
  for (int i=0; i<respuestas.length; i++)
  {
    puntos = puntos + puntosPregunta(respuestas[i], claves[i]);
  }
  return puntos;
}
```
